Google Summer of Code 2025 - OpenStreetMap Project Proposal

Raj Rajeshwar Singh Bisen

March 26, 2025

General Information

- **Name:** Raj Rajeshwar Singh Bisen
- **OSM Account Name:** raj_rajeshwar_singh_bisen
- **Current Occupation:** Bachelor of Technology in Computer Science and Engineering at SNU, GN, India
- **Contact Information:**
  - Email: thatonebipanda@gmail.com
  - GitHub: AnarchistHoneybun

Technical Background

Programming and Technical Skills

Languages

- **Advanced Level:** C, Java, Python, Rust, HTML/CSS, Bash
- **Proficient Level:** PowerShell, C++, JavaScript, TypeScript, LaTeX
- **Functional Understanding:** Haskell, Solidity, Zig, Go

Software and Tools

- **Development Environments:** VSCode, JetBrains IDEs, Visual Studio
- **Version Control:** Git, GitHub
- **Operating Systems:** Arch Linux, Windows

Libraries and Frameworks

- **Web Development:** React, Next.js
- **Scientific Computing:** NumPy, SciPy, Matplotlib, TensorFlow
- **Rust Ecosystem:** Ratatui, Tokio, Charm CLI

Databases and Backend Technologies

- **Relational Databases:** PostgreSQL, MariaDB, MySQL
- **Cloud Databases:** Supabase, Firebase
- **Database Management:** Advanced querying, database design, and optimization

Project-Specific Skills

Considering the Temporary Road Closures Database and API project, I have particularly relevant skills:

- Extensive experience with PostgreSQL, MariaDB, and other SQL databases
- Proficiency in developing robust APIs using Python, Node, and Rust
- Experience with cloud database solutions like Supabase and Firebase
- Capable of creating user interfaces for data entry and visualization

Previous Programming Projects

CredHealthBackend: Blockchain-Powered Dynamic Health Insurance System

- **Purpose:** A hackathon project integrating health metrics (sleep, walking, and food tracking) with insurance policies, dynamically adjusting premiums based on user habits via blockchain smart contracts.
- **Role:** Team of 3; I led the blockchain backend development and co-designed the mobile app frontend.
- **Technical Implementation:**
  - **Blockchain:**
    - Developed Solidity smart contracts to mint insurance policies as NFTs, with functions to adjust health scores and premiums.
    - Deployed a local testnet using Ganache; contracts handled real-time health data fluctuations.
  - **Backend:**
    - Built a Node.js API layer to bridge the React Native frontend (CredHealth repo) and blockchain.
    - Stored food images in AWS S3 and processed them with Python (quantity/quality analysis).
  - **Security:**
    - Ensured data integrity via blockchain immutability; limited PI health information to user device to minimize risk of leak.
- **Challenges & Solutions:**
  - **Gas Fees & Scalability:** Optimized contract functions to minimize transactions during testing.

- **Outcome:**
  - Functional prototype deployed on a local network; demoed on-device with dynamic premium adjustments.
  - Won special mention (4th place overall) in the software projects category.
- **Technologies:** Solidity, Ganache, Node.js, React Native, AWS S3, Python (OpenCV/TensorFlow for image analysis), Git.

Kupyna Hash Function Implementation for RustCrypto

- **Purpose:** Implemented Kupyna (DSTU 7564:2014), a Ukrainian national standard hash function, as part of RustCrypto's mission to provide pure-Rust cryptographic primitives. Based on the IACR specification, this contribution expands RustCrypto's algorithm support for developers requiring standards-compliant hashing.
- **Role:** One of two contributors; proposed and developed the implementation with my collaborator, later adapting it to match RustCrypto's trait system for seamless integration.
- **Technical Implementation:**
  - **Algorithm:**
    - Translated the Kupyna specification into idiomatic Rust.
    - Aligned with RustCrypto's `Digest` trait for interoperability with other crates.
  - **Testing:**
    - Verified correctness against official test vectors from the IACR paper.
    - Integrated into RustCrypto's CI/CD pipeline for automated testing and other checks.
  - **Performance:**
    - Actively optimizing throughput (e.g., leveraging new Galois Multiplication methods and working dircetly on 64 bit chunks in ongoing work).
- **Challenges & Solutions:**
  - **Trait Integration:** Studied RustCrypto's existing hash implementations (e.g., Groestl) to ensure API consistency.
- **Outcome:**
  - Successfully merged into RustCrypto/hashes and published as part of the official suite.
  - Downstream-ready for use in authentication, blockchain, or other cryptographic applications.
- **Technologies:** Rust, GitHub Actions (CI/CD).

Contributions to the Target Project

As per the GSoC guidelines, I contributed to the **iD editor** (a core OpenStreetMap web application) to demonstrate my ability to work with OSM's ecosystem. While my proposed GSoC project involves creating a new system for Temporary Road Closures (with no existing repository), my Pull Request #10903 to iD showcases relevant technical and collaborative skills for web-based OSM tools.

- **Feature Implementation:** Addressed Issue #10870 by implementing an auto toggle for highlighting unsaved changes on iD editor startup. This helps users quickly identify pending edits from their previous session, improving workflow efficiency.
- **Technical Scope:**
  - Modified the JavaScript codebase to persist and render unsaved change markers.
  - Leveraged iD's state management system to track edits across sessions.
- **Development Process:**
  - Set up the iD project locally, navigating its build system and architecture.
  - Analyzed the editor's event-driven workflow to integrate the feature without disrupting core functionality.
- **Relevance to Proposed Project:** Though my GSoC project would be new, this contribution demonstrates:
  - Experience with web apps.
  - Ability to work within OSM's technical and community standards.
  - Skill in enhancing user-facing tools—critical for the planned Road Closures API's frontend components.

This contribution underscores my readiness to develop web-based tools for OSM, even as I transition to a new codebase for my GSoC project.

Project Proposal

Project Title

Temporary road closures database and API

Project Overview

[concise summary of what's intended to be accomplished]

Problem Statement

[articulate problem and issues it's meant to address]

Proposed Solution

[Describe plan in detail to solve the problem. Include:

- Technical approach
- Tools and technologies
- Integration
- Design considerations or limitations

]

Expected Outcomes

- **Core Deliverables:** [essenatial components]
- **Stretch Goals:** [Additional features]
- **Fallback Plan:** [contingencies]

Technical Implementation Details

[specific technical details about implementation plan. include:

- System architecture
- API designs
- Database schemas
- Algorithms
- Dependencies and integration points

diagrams(?).]

Learning Objectives

[knowledge goals, alignment with learning objectives etc]

Project Timeline

Availability

- **Planned Vacations:** [vacations or time off]
- **Concurrent Commitments:** [Classes, employment, or other obligations during GSoC]
- **Weekly Availability:** [weekly/daily availability]

Project Schedule

[detailed week-by-week schedule showing:

- Tasks to be completed
- Major milestones
- Testing periods
- Documentation phases

1-2 weeks per step]

Week	Tasks	Hours
1	[List specific tasks for week 1]	[Hours]
2	[List specific tasks for week 2]	[Hours]
...	...	...

Total Commitment

[verify hours adding up.]

Communication Plan

Meeting Schedule

[regular meeting schedule with mentor(s).]

Progress Reporting

[weekly/daily updates, blogs(?), github projects(?)]

Use of AI Tools

[disclose AI tool usage, explain output verification and scope and method of use]

Additional Information

[Include any other information that might be relevant to application.]